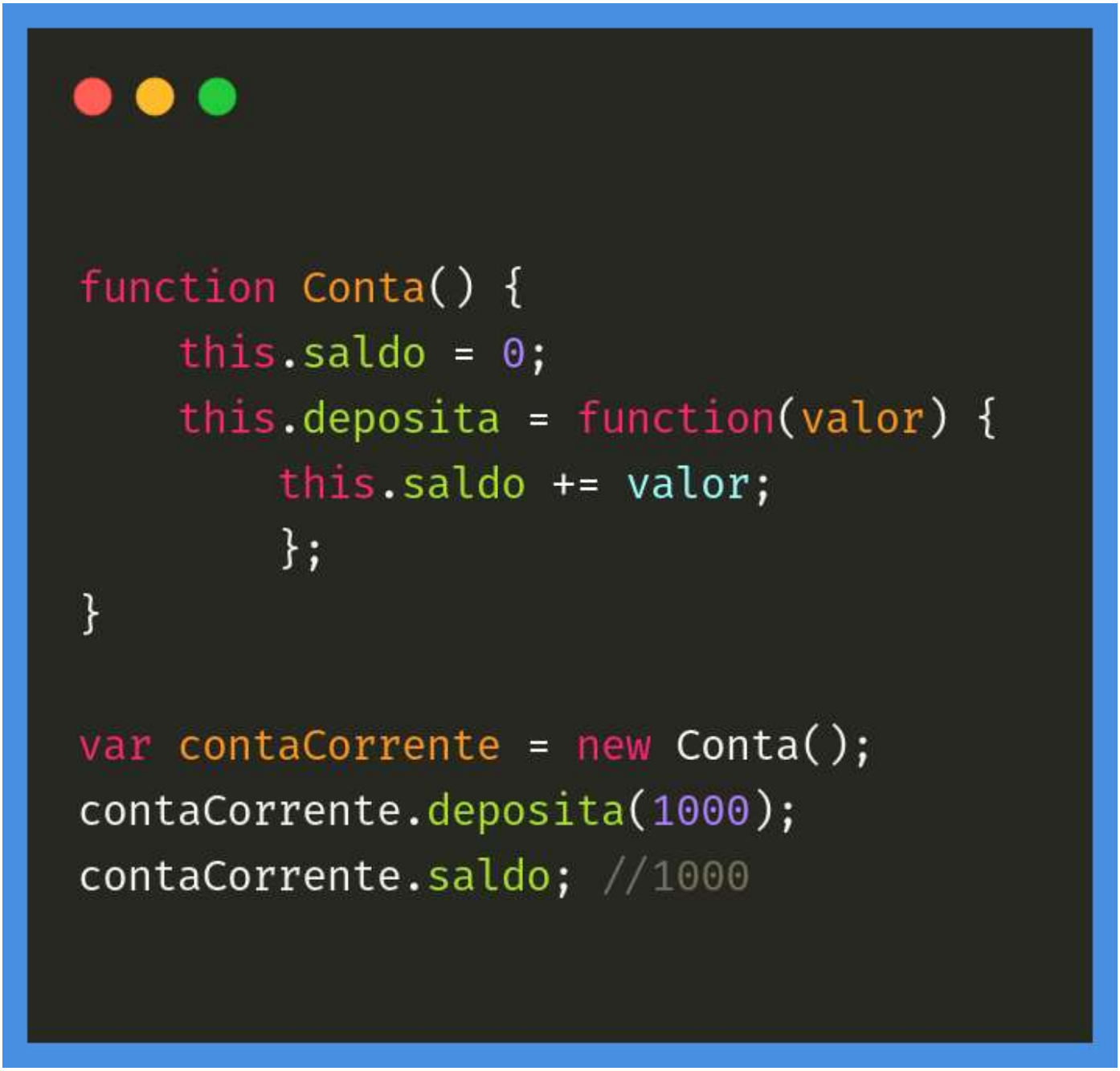
Herencia en JavaScript

alexandre-aquiles

06/04/2022



Tenemos un constructor `Cuenta` que establece un atributo `saldo` y un método `deposita`:

```
    function Cuenta() {
    this.balance = 0;
     this.deposita = function(valor) {
         this.balance += valor
         };
}
var cuentaCorriente = new Cuenta();
```

```
cuentaCorriente.deposita(1000);
cuentaCorriente.balance; //1000
```

¡Genial, funciona!

Ahora, nuestro cliente necesita una cuenta de ahorros: un tipo de cuenta que actualiza el saldo según un índice.

Es muy fácil de hacer: sólo hay que crear un constructor `CuentaAhorros`, copiar y pegar todo el código de `Cuenta` y definir un nuevo método `actualiza`:

```
function CuentaAhorros() {
    //copia de la cuenta
    this.balance = 0; this.deposita = function(valor) { this.balance += valor;
    //método adicional...
    this.actualiza = function(índice) { this.balance += this.balance * índice;

var cuentaAhorros = new cuentaAhorros();
 cuentaAhorros.deposita(1000);
 cuentaAhorros.balance; //1000
 cuentaAhorros.actualiza(0.05);
 cuentaAhorros.balance; //1050
```

Bueno, pero **copiar y pegar es malo**. Cada vez que cambiamos o ampliamos `Cuenta`, tenemos que acordarnos de copiarla en `CuentaAhorros`.

Lo ideal sería **reutilizar** el código de `Cuenta`, haciendo de `CuentaAhorros` un **tipo especial** de `Cuenta`, con todo lo que tiene `Cuenta` más el método `actualiza`.

En los lenguajes orientados a objetos, esta idea de crear un tipo especial basado en otro se llama ***Herencia***. El JavaScript es un linguaje orientado a objetos y tiene soporte de herencia.

La mayoría de los lenguajes tienen clases, con alguna forma de extenderlas. Pero JavaScript no es un lenguaje "clásico" y la herencia en JavaScript es un poco diferente,

basada en **prototipos**.

Primero, creo el constructor de `CuentaAhorro` sin las propiedades que quiero heredar, sólo con el método extra:

```
function CuentaAhorros() {
    //no necesito copiar de Cuenta...
     this.actualiza = function(índice) {
         this.balance += this.balance * índice
        };
    }
```

Entonces, utilizando el `prototype` de `CuentaAhorros`, digo que quiero copiar todo lo que hay en `Cuenta` en `CuentaAhorros`.

```
CuentaAhorros.prototype = new Cuenta();
CuentaAhorros.prototype.constructor = CuentaAhorros;
```

De hecho hay otras formas de copiar las propiedades de `Cuenta` en el `prototype` de `CuentaAhorros`. La forma anterior es la más común.

Por último, podemos crear una cuenta de ahorros y utilizar todo lo de una cuenta, más la actualización:

```
var cuentaAhorros = new CuentaAhorro();
cuentaAhorros.deposita(1000);
cuentaAhorros.actualiza(0.05);
cuentaAhorros.balance; //1050
```

En la última versión de JavaScript, *EcmaScript 2018*, se puede definir la herencia mediante la palabra clave `extends`, presente en muchos otros lenguajes:

```
class Cuenta { constructor() {
    this.balance = 0; }
    deposita(valor) { this.balance += valor; }
```

```
class CuentaAhorros extends Cuenta {
    actualiza(índice) { this.balance += this.balance * índice; }
```

El lenguaje Javascript tiene una gran cantidad de funciones potentes! Formación Front-end de Alura Latam.

Este articulo fue adecuado para Alura Latam por: Marianna Costa

Puedes leer también:

- ¿Qué es DOM?
- Como usar el terminal integrado de Visual Studio Code

ARTÍCULOS DE TECNOLOGÍA > FRONT END

- Estudia las 24 horas, los 7 días de la semana
- Foro y comunidad exclusiva para resolver tus dudas
- Acceso a todo el contenido de la plataforma por 6 meses

¡QUIERO EMPEZAR A ESTUDIAR!

Paga en moneda local en los siguientes países

ANUAL

US$79,90

un solo pago de US$79,90

- 218 cursos
- Videos y actividades 100% en Español
- Certificado de participación
- Estudia las 24 horas, los 7 días de la semana

Nuevos cursos
cada semana

NAVEGACIÓN

PLANES
INSTRUCTORES
BLOG
POLÍTICA DE PRIVACIDAD
TÉRMINOS DE USO
SOBRE NOSOTROS
PREGUNTAS FRECUENTES

¡CONTÁCTANOS!

¡QUIERO ENTRAR EN CONTACTO!

BLOG

PROGRAMACIÓN
FRONT END
DATA SCIENCE
INNOVACIÓN Y GESTIÓN
DEVOPS

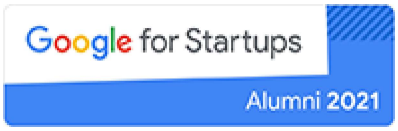
AOVS Sistemas de Informática S.A
CNPJ 05.555.382/0001-33

SÍGUENOS EN NUESTRAS REDES SOCIALES

ALIADOS

Empresa participante do SCALE UP ENDEAVOR

En Alura somos unas de las Scale-Ups seleccionadas por Endeavor, programa de aceleración de las empresas que más crecen en el país.



Fuimos unas de las 7 startups seleccionadas por Google For Startups en participar del programa Growth Academy en 2021

POWERED BY

CURSOS

Cursos de Programación
Lógica de Programación | Java

Cursos de Front End
HTML y CSS | JavaScript | React

Cursos de Data Science
Data Science | Machine Learning | Excel | Base de Datos | Data Visualization | Estadística

Cursos de DevOps
Docker | Linux

Cursos de Innovación y Gestión
Productividad y Calidad de Vida | Transformación Ágil | Marketing Analytics | Liderazgo y Gestión de Equipos | Startups y Emprendimiento